APPENDIX B: Portions of the MPEG-4 Standard

Object based coding syntax

Video object

A *video object* in a scene is an entity that a user is allowed to access (seek, browse) and manipulate (cut and paste). The instances of video objects at a given time are called *video object planes* (VOPs). The encoding process generates a coded representation of a VOP as well as composition information necessary for display. Further, at the decoder, a user may interact with and modify the composition process as needed.

The full syntax allows coding of rectangular as well as arbitrarily shaped video objects in a scene. Furthermore, the syntax supports both nonscalable coding and scalable coding. Thus it becomes possible to handle normal scalabilities as well as object based scalabilities. The scalability syntax enables the reconstruction of useful video from pieces of a total bitstream. This is achieved by structuring the total bitstream in two or more layers, starting from a standalone base layer and adding a number of enhancement layers. The base layer can be coded using a non-scalable syntax, or in the case of picture based coding, even using a syntax of a different video coding standard.

To ensure the ability to access individual objects, it is necessary to achieve a coded representation of its shape. A natural video object consists of a sequence of 2D representations (at different points in time) referred to here as VOPs. For efficient coding of VOPs, both temporal redundancies as well as spatial redundancies are exploited. Thus a coded representation of a VOP includes representation of its shape, its motion and its texture.

Face object

A 3D (or 2D) *face object* is a representation of the human face that is structured for portraying the visual manifestations of speech and facial expressions adequate to achieve visual speech intelligibility and the recognition of the mood of the speaker. A face object is animated by a stream of *face animation parameters (FAP)* encoded for low-bandwidth transmission in broadcast (one-to-many) or dedicated interactive (point-to-point) communications. The FAPs manipulate key feature control points in a mesh model of the face to produce animated visemes for the mouth (lips, tongue, teeth), as well as animation of the head and facial features like the eyes. FAPs are quantized with careful consideration for the limited movements of facial features, and then prediction errors are calculated and coded arithmetically. The remote manipulation of a face model in a terminal with FAPs can accomplish lifelike visual scenes of the speaker in real-time without sending pictorial or video details of face imagery every frame.

A simple streaming connection can be made to a decoding terminal that animates a default face model. A more complex session can initialize a custom face in a more capable terminal by downloading *face definition parameters (FDP)* from the encoder. Thus specific background images, facial textures, and head geometry can be portrayed. The composition of specific backgrounds, face 2D/3D meshes, texture attribution of the mesh, etc. is described in ISO/IEC 14496-1. The FAP stream for a given user can be generated at the user's terminal from video/audio, or from text-to-speech. FAPs can be encoded at bitrates up to 2-3kbit/s at necessary speech rates. Optional temporal DCT coding provides further compression efficiency in exchange for delay. Using the facilities of ISO/IEC 14496-1, a composition of the animated face model and synchronized, coded speech audio (low-bitrate speech coder or text-to-speech) can provide an integrated low-bandwidth audio/visual speaker for broadcast applications or interactive conversation.

Limited scalability is supported. Face animation achieves its efficiency by employing very concise motion animation controls in the channel, while relying on a suitably equipped terminal for rendering of moving 2D/3D faces with non-normative models held in local memory. Models stored and updated for rendering in the terminal can be simple or complex. To support speech intelligibility, the normative specification of FAPs intends for their selective or complete use as signaled by the encoder. A masking scheme provides for selective transmission of FAPs according to what parts of the face are naturally active from moment to moment. A further control in the FAP stream allows face animation to be suspended while leaving face features in the terminal in a defined quiescent state for higher overall efficiency during multi-point connections.

The Face Animation specification is defined in ISO/IEC 14496-1 and this part of ISO/IEC 14496. This clause is intended to facilitate finding various parts of specification. As a rule of thumb, FAP specification is found in the part 2, and FDP specification in the part 1. However, this is not a strict rule. For an overview of FAPs and their interpretation, read subclauses "6.1.5.2 Facial animation parameter set", "6.1.5.3 Facial animation parameter units", "6.1.5.4 Description of a neutral face" as well as the Table C-1. The viseme parameter is documented in subclause "7.12.3 Decoding of the viseme parameter fap 1" and the Table C-5 in annex C. The expression parameter is

Motion representation - macroblocks

The choice of 16×16 blocks (referred to as macroblocks) for the motion-compensation unit is a result of the trade-off between the coding gain provided by using motion information and the overhead needed to represent it. Each macroblock can further be subdivided to 8×8 blocks for motion estimation and compensation depending on the overhead that can be afforded.

Depending on the type of the macroblock, motion vector information and other side information is encoded with the compressed prediction error in each macroblock. The motion vectors are differenced with respect to a prediction value and coded using variable length codes. The maximum length of the motion vectors allowed is decided at the encoder. It is the responsibility of the encoder to calculate appropriate motion vectors. The specification does not specify how this should be done.

Spatial redundancy reduction

Both source VOPs and prediction errors VOPs have significant spatial redundancy. This part of ISO/IEC 14496 uses a block-based DCT method with optional visually weighted quantisation, and run-length coding. After motion compensated prediction or interpolation, the resulting prediction error is split into 8×8 blocks. These are transformed into the DCT domain where they can be weighted before being quantised. After quantisation many of the DCT coefficients are zero in value and so two-dimensional run-length and variable length coding is used to encode the remaining DCT coefficients efficiently.

Chrominance formats

This part of ISO/IEC 14496 currently supports the 4:2:0 chrominance format.

Pixel depth

This part of ISO/IEC 14496 supports pixel depths between 4 and 12 bits in luminance and chrominance planes.

Generalized scalability

The scalability tools in this part of ISO/IEC 14496 are designed to support applications beyond that supported by single layer video. The major applications of scalability include internet video, wireless video, multi-quality video services, video database browsing etc. In some of these applications, either normal scalabilities on picture basis such as those in ISO/IEC 13818-2 may be employed or object based scalabilities may be necessary; both categories of scalability are enabled by this part of ISO/IEC 14496.

Although a simple solution to scalable video is the simulcast technique that is based on transmission/storage of multiple independently coded reproductions of video, a more efficient alternative is scalable video coding, in which the bandwidth allocated to a given reproduction of video can be partially re-utilised in coding of the next reproduction of video. In scalable video coding, it is assumed that given a coded bitstream, decoders of various complexities can decode and display appropriate reproductions of coded video. A scalable video encoder is likely to have increased complexity when compared to a single layer encoder. However, this part of ISO/IEC 14496 provides several different forms of scalabilities that address non-overlapping applications with corresponding complexities.

The basic scalability tools offered are temporal scalability and spatial scalability. Moreover, combinations of these basic scalability tools are also supported and are referred to as hybrid scalability. In the case of basic scalability, two layers of video referred to as the lower layer and the enhancement layer are allowed, whereas in hybrid scalability up to four layers are supported.

Object based Temporal scalability

Temporal scalability is a tool intended for use in a range of diverse video applications from video databases, internet video, wireless video and multiview/stereoscopic coding of video. Furthermore, it may also provide a migration path from current lower temporal resolution video systems to higher temporal resolution systems of the future.

Temporal scalability involves partitioning of VOPs into layers, where the lower layer is coded by itself to provide the basic temporal rate and the enhancement layer is coded with temporal prediction with respect to the lower layer. These layers when decoded and temporally multiplexed yield full temporal resolution. The lower temporal resolution systems may only decode the lower layer to provide basic temporal resolution whereas enhanced systems of the future may support both layers. Furthermore, temporal scalability has use in bandwidth constrained networked applications where adaptation to frequent changes in allowed throughput are necessary. An additional advantage of temporal scalability is its ability to provide resilience to transmission errors as the more important data of the lower layer can be sent over a channel with better error performance, whereas the less critical enhancement layer can be sent over a channel with poor error performance. Object based temporal scalability can also be employed to allow graceful control of picture quality by controlling the temporal rate of each video object under the constraint of a given bit-budget.

Spatial scalability

Spatial scalability is a tool intended for use in video applications involving multi quality video services, video database browsing, internet video and wireless video, i.e., video systems with the primary common feature that a minimum of two layers of spatial resolution are necessary. Spatial scalability involves generating two spatial resolution video layers from a single video source such that the lower layer is coded by itself to provide the basic spatial resolution and the enhancement layer employs the spatially interpolated lower layer and carries the full spatial resolution of the input video source.

An additional advantage of spatial scalability is its ability to provide resilience to transmission errors as the more important data of the lower layer can be sent over a channel with better error performance, whereas the less critical enhancement layer data can be sent over a channel with poor error performance. Further, it can also allow interoperability between various standards.

Hybrid scalability

There are a number of applications where neither the temporal scalability nor the spatial scalability may offer the necessary flexibility and control. This may necessitate use of temporal and spatial scalability simultaneously and is referred to as the hybrid scalability. Among the applications of hybrid scalability are wireless video, internet video, multiviewpoint/stereoscopic coding etc.

Error Resilience

This part of ISO/IEC 14496 provides error robustness and resilience to allow accessing of image or video information over a wide range of storage and transmission media. The error resilience tools developed for this part of ISO/IEC 14496 can be divided into three major categories. These categories include synchronization, data recovery, and error concealment. It should be noted that these categories are not unique to this part of ISO/IEC 14496, and have been used elsewhere in general research in this area. It is, however, the tools contained in these categories that are of interest, and where this part of ISO/IEC 14496 makes its contribution to the problem of error resilience.

Patents

The International Organization for Standarization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this part of ISO/IEC 14496 may involve the use of patents concerning the coded representation of picture information given in Annex H.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statements of the holders of these patent rights are registered with ISO and IEC. Information may be obtained from the patent offices of the organizations listed in Annex H.

Attention is drawn to the possibility that some of the elements of this part of ISO/IEC 14496 may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

- IEEE Standard Specifications for the Implementations of 8 by 8 Inverse Discrete Cosine Transform, IEEE Std 1180-1990, December 6, 1990.
- IEC Publication 908:1987, *CD Digital Audio System.*
- IEC Publication 461:1986, *Time and control code for video tape recorder.*
- ITU-T Recommendation H.261 (Formerly CCITT Recommendation H.261), *Codec for audiovisual services at px64 kbit/s.*
- ITU-T Recommendation H.263, *Video Coding for Low Bitrate Communication.*

3 Definitions

3.1 **AC coefficient**: Any DCT coefficient for which the frequency in one or both dimensions is non-zero.

3.2 **B-VOP; bidirectionally predictive-coded video object plane (VOP)**: A VOP that is coded using motion compensated prediction from past and/or future reference VOPs.

3.3 **backward compatibility**: A newer coding standard is backward compatible with an older coding standard if decoders designed to operate with the older coding standard are able to continue to operate by decoding all or part of a bitstream produced according to the newer coding standard.

3.4 **backward motion vector**: A motion vector that is used for motion compensation from a reference VOP at a later time in display order.

3.5 **backward prediction**: Prediction from the future reference VOP.

3.6 **base layer**: An independently decodable layer of a scalable hierarchy.

3.7 **binary alpha block**: A block of size 16x16 pels, colocated with macroblock, representing shape information of the binary alpha map; it is also referred to as a bab.

3.8 **binary alpha map**: A 2D binary mask used to represent the shape of a video object such that the pixels that are opaque are considered as part of the object where as pixels that are transparent are not considered to be part of the object.

3.9 **bitstream; stream**: An ordered series of bits that forms the coded representation of the data.

3.10 **bitrate**: The rate at which the coded bitstream is delivered from the storage medium or network to the input of a decoder.

3.11 **block**: An 8-row by 8-column matrix of samples, or 64 DCT coefficients (source, quantised or dequantised).

3.12 **byte aligned**: A bit in a coded bitstream is byte-aligned if its position is a multiple of 8-bits from the first bit in the stream.

3.13 **byte**: Sequence of 8-bits.

3.14 **context based arithmetic encoding**: The method used for coding of binary shape; it is also referred to as cae.

3.15 **channel**: A digital medium or a network that stores or transports a bitstream constructed according to ISO/IEC 14496.

3.16 **chrominance format**: Defines the number of chrominance blocks in a macroblock.

3.41 **display order**: The order in which the decoded pictures are displayed. Normally this is the same order in which they were presented at the input of the encoder.

3.42 **editing**: The process by which one or more coded bitstreams are manipulated to produce a new coded bitstream. Conforming edited bitstreams must meet the requirements defined in this part of ISO/IEC 14496.

3.43 **encoder**: An embodiment of an encoding process.

3.44 **encoding (process)**: A process, not specified in this part of ISO/IEC 14496, that reads a stream of input pictures or audio samples and produces a valid coded bitstream as defined in this part of ISO/IEC 14496.

3.45 **enhancement layer**: A relative reference to a layer (above the base layer) in a scalable hierarchy. For all forms of scalability, its decoding process can be described by reference to the lower layer decoding process and the appropriate additional decoding process for the enhancement layer itself.

3.46 **face animation parameter units, FAPU**: Special normalized units (e.g. translational, angular, logical) defined to allow interpretation of FAPs with any facial model in a consistent way to produce reasonable results in expressions and speech pronunciation.

3.47 **face animation parameters, FAP**: Coded streaming animation parameters that manipulate the displacements and angles of face features, and that govern the blending of visemes and face expressions during speech.

3.48 **face animation table, FAT**: A downloadable function mapping from incoming FAPs to feature control points in the face mesh that provides piecewise linear weightings of the FAPs for controlling face movements.

3.49 **face calibration mesh**: Definition of a 3D mesh for calibration of the shape and structure of a baseline face model.

3.50 **face definition parameters, FDP**: Downloadable data to customize a baseline face model in the decoder to a particular face, or to download a face model along with the information about how to animate it. The FDPs are normally transmitted once per session, followed by a stream of compressed FAPs. FDPs may include feature points for calibrating a baseline face, face texture and coordinates to map it onto the face, animation tables, etc.

3.51 **face feature control point**: A normative vertex point in a set of such points that define the critical locations within face features for control by FAPs and that allow for calibration of the shape of the baseline face.

3.52 **face interpolation transform, FIT**: A downloadable node type defined in ISO/IEC 14496-1 for optional mapping of incoming FAPs to FAPs before their application to feature points, through weighted rational polynomial functions, for complex cross-coupling of standard FAPs to link their effects into custom or proprietary face models.

3.53 **face model mesh**: A 2D or 3D contiguous geometric mesh defined by vertices and planar polygons utilizing the vertex coordinates, suitable for rendering with photometric attributes (e.g. texture, color, normals).

3.54 **feathering**: A tool that tapers the values around edges of binary alpha mask for composition with the background.

3.55 **flag**: A one bit integer variable which may take one of only two values (zero and one).

3.56 **forbidden**: The term "forbidden" when used in the clauses defining the coded bitstream indicates th the value shall never be used. This is usually to avoid emulation of start codes.

3.57 **forced updating**: The process by which macroblocks are intra-coded from time-to-time to ensure th mismatch errors between the inverse DCT processes in encoders and decoders cannot build u excessively.

3.58 **forward compatibility**: A newer coding standard is forward compatible with an older coding standard decoders designed to operate with the newer coding standard are able to decode bitstreams of th older coding standard.

3.59 **forward motion vector**: A motion vector that is used for motion compensation from a reference fram VOP at an earlier time in display order.

3.60 **forward prediction**: Prediction from the past reference VOP.

3.61 **frame**: A frame contains lines of spatial information of a video signal. For progressive video, these lines contain samples starting from one time instant and continuing through successive lines to the bottom of the frame.

3.62 **frame period**: The reciprocal of the frame rate.

3.63 **frame rate**: The rate at which frames are be output from the composition process.

3.64 **future reference VOP**: A future reference VOP is a reference VOP that occurs at a later time than the current VOP in display order.

3.65 **VOP reordering**: The process of reordering the reconstructed VOPs when the decoding order is different from the composition order for display. VOP reordering occurs when B-VOPs are present in a bitstream. There is no VOP reordering when decoding low delay bitstreams.

3.66 **hybrid scalability**: Hybrid scalability is the combination of two (or more) types of scalability.

3.67 **interlace**: The property of conventional television frames where alternating lines of the frame represent different instances in time. In an interlaced frame, one of the field is meant to be displayed first. This field is called the first field. The first field can be the top field or the bottom field of the frame.

3.68 **I-VOP; intra-coded VOP**: A VOP coded using information only from itself.

3.69 **intra coding**: Coding of a macroblock or VOP that uses information only from that macroblock or VOP.

3.70 **intra shape coding**: Shape coding that does not use any temporal prediction.

3.71 **inter shape coding**: Shape coding that uses temporal prediction.

3.72 **level**: A defined set of constraints on the values which may be taken by the parameters of this part of ISO/IEC 14496 within a particular profile. A profile may contain one or more levels. In a different context, level is the absolute value of a non-zero coefficient (see "run").

3.73 **layer**: In a scalable hierarchy denotes one out of the ordered set of bitstreams and (the result of) its associated decoding process.

3.74 **layered bitstream**: A single bitstream associated to a specific layer (always used in conjunction with layer qualifiers, e. g. "enhancement layer bitstream").

3.75 **lower layer**: A relative reference to the layer immediately below a given enhancement layer (implicitly including decoding of *all* layers below this enhancement layer).

3.95 **profile**: A subset of the syntax of this part of ISO/IEC 14496, defined in terms of Visual Object Types.

3.96 **progressive**: The property of film frames where all the samples of the frame represent the sam instances in time.

3.97 **quantisation matrix**: A set of sixty-four 8-bit values used by the dequantiser.

3.98 **quantised DCT coefficients**: DCT coefficients before dequantisation. A variable length code representation of quantised DCT coefficients is transmitted as part of the coded video bitstream.

3.99 **quantiser scale**: A scale factor coded in the bitstream and used by the decoding process to scale the dequantisation.

3.100 **random access**: The process of beginning to read and decode the coded bitstream at an arbitrary point.

3.101 **reconstructed VOP**: A reconstructed VOP consists of three matrices of 8-bit numbers representing the luminance and two chrominance signals. It is obtained by decoding a coded VOP.

3.102 **reference VOP**: A reference VOP is a reconstructed VOP that was coded in the form of a coded I-VOP or a coded P-VOP. Reference VOPs are used for forward and backward prediction when P-VOPs and B-VOPs are decoded.

3.103 **reordering delay**: A delay in the decoding process that is caused by VOP reordering.

3.104 **reserved**: The term "reserved" when used in the clauses defining the coded bitstream indicates that the value may be used in the future for ISO/IEC defined extensions.

3.105 **scalable hierarchy**: coded video data consisting of an ordered set of more than one video bitstream.

3.106 **scalability**: Scalability is the ability of a decoder to decode an ordered set of bitstreams to produce a reconstructed sequence. Moreover, useful video is output when subsets are decoded. The minimum subset that can thus be decoded is the first bitstream in the set which is called the base layer. Each of the other bitstreams in the set is called an enhancement layer. When addressing a specific enhancement layer, "lower layer" refers to the bitstream that precedes the enhancement layer.

3.107 **side information**: Information in the bitstream necessary for controlling the decoder.

3.108 **run**: The number of zero coefficients preceding a non-zero coefficient, in the scan order. The absolute value of the non-zero coefficient is called "level".

3.109 **S-VOP**: A picture that is coded using information obtained by warping whole or part of a static sprite.

3.110 **saturation**: Limiting a value that exceeds a defined range by setting its value to the maximum or minimum of the range as appropriate.

3.111 **source; input**: Term used to describe the video material or some of its attributes before encoding.

3.112 **spatial prediction**: prediction derived from a decoded frame of the reference layer decoder used in spatial scalability.

3.113 **spatial scalability**: A type of scalability where an enhancement layer also uses predictions from sample data derived from a lower layer without using motion vectors. The layers can have different VOP sizes or VOP rates.

3.114 **static sprite**: The luminance, chrominance and binary alpha plane for an object which does not vary in time.

5.2.6 Definition of transparent_mb() function

The function transparent_mb() returns 1 if the current macroblock consists only of transparent pixels. Otherwise returns 0.

5.2.7 Definition of transparent_block() function

The function transparent_block(j) returns 1 if the 8x8 with index j consists only of transparent pixels. Otherwise returns 0. The index value for each block is defined in Figure 6-5.

5.3 Reserved, forbidden and marker_bit

The terms "reserved" and "forbidden" are used in the description of some values of several fields in the coded bitstream.

The term "reserved" indicates that the value may be used in the future for ISO/IEC defined extensions.

The term "forbidden" indicates a value that shall never be used (usually in order to avoid emulation of start codes).

The term "marker_bit" indicates a one bit integer in which the value zero is forbidden (and it therefore shall have the value '1'). These marker bits are introduced at several points in the syntax to avoid start code emulation.

The term "zero_bit" indicates a one bit integer with the value zero.

5.4 Arithmetic precision

In order to reduce discrepancies between implementations of this part of ISO/IEC 14496, the following rules for arithmetic operations are specified.

(a) Where arithmetic precision is not specified, such as in the calculation of the IDCT, the precision shall be sufficient so that significant errors do not occur in the final integer values.

(b) Where ranges of values are given, the end points are included if a square bracket is present, and excluded if a round bracket is used. For example, [a , b) means from a to b, including a but excluding b.

6 Visual bitstream syntax and semantics

6.1 Structure of coded visual data

Coded visual data can be of several different types, such as video data, still texture data, 2D mesh data or facial animation parameter data.

Synthetic objects and their attribution are structured in a hierarchical manner to support both bitstream scalability and object scalability. ISO/IEC 14496-1 of the specification provides the approach to spatial-temporal scene composition including normative 2D/3D scene graph nodes and their composition supported by Binary Interchange Format Specification. At this level, synthetic and natural object composition relies on ISO/IEC 14496-1 with subsequent (non-normative) rendering performed by the application to generate specific pixel-oriented views of the models.

Coded video data consists of an ordered set of video bitstreams, called layers. If there is only one layer, the coded video data is called non-scalable video bitstream. If there are two layers or more, the coded video data is called a scalable hierarchy.

One of the layers is called base layer, and it can always be decoded independently. Other layers are called enhancement layers, and can only be decoded together with the lower layers (previous layers in the ordered set), starting with the base layer. The multiplexing of these layers is discussed in ISO/IEC 14496-1. The base layer of a scalable set of streams can be coded by other standards. The Enhancement layers shall conform to this part of ISO/IEC 14496. In general the visual bitstream can be thought of as a syntactic hierarchy in which syntactic structures contain one or more subordinate structures.

Visual texture, referred to herein as still texture coding, is designed for maintaining high visual quality in the transmission and rendering of texture under widely varied viewing conditions typical of interaction with 2D/3D synthetic scenes. Still texture coding provides for a multi-layer representation of luminance, color and shape. This supports progressive transmission of the texture for image build-up as it is received by a terminal. Also supported is the downloading of the texture resolution hierarchy for construction of image pyramids used by 3D graphics APIs. Quality and SNR scalability are supported by the structure of still texture coding.

Coded mesh data consists of a single non-scalable bitstream. This bitstream defines the structure and motion of a 2D mesh object. Texture that is to be mapped onto the mesh geometry is coded separately.

Coded face animation parameter data consists of one non-scaleable bitstream. It defines the animation of the facemodel of the decoder. Face animation data is structured as standard formats for downloadable models and their animation controls, and a single layer of compressed face animation parameters used for remote manipulation of the face model. The face is a node in a scene graph that includes face geometry ready for rendering. The shape, texture and expressions of the face are generally controlled by the bitstream containing instances of Facial Definition Parameter (FDP) sets and/or Facial Animation Parameter (FAP) sets. Upon initial or baseline construction, the face object contains a generic face with a neutral expression. This face can receive FAPs from the bitstream and be subsequently rendered to produce animation of the face. If FDPs are transmitted, the generic face is transformed into a particular face of specific shape and appearance. A downloaded face model via FDPs is a scene graph for insertion in the face node.

6.1.1 Visual object sequence

Visual object sequence is the highest syntactic structure of the coded visual bitstream.

A visual object sequence commences with a visual_object_sequence_start_code which is followed by one or more visual objects coded concurrently. The visual object sequence is terminated by a visual_object_sequence_end_code.

6.1.2 Visual object

A visual object commences with a visual_object_start_code, is followed by profile and level identification, and a visual object id, and is followed by a video object, a still texture object, a mesh object, or a face object.

6.1.3 Video object

A video object commences with a video_object_start_code, and is followed by one or more video object layers.

6.1.3.1 Progressive and interlaced sequences

This part of ISO/IEC 14496 deals with coding of both progressive and interlaced sequences.

The sequence, at the output of the decoding process, consists of a series of reconstructed VOPs separated in time and are readied for display via the compositor.

6.1.3.2 Frame

A frame consists of three rectangular matrices of integers; a luminance matrix (Y), and two chrominance matrices (Cb and Cr).

6.1.3.3 VOP

A reconstructed VOP is obtained by decoding a coded VOP. A coded VOP may have been derived from either a progressive or interlaced frame.

6.1.3.4 VOP types

There are four types of VOPs that use different coding methods:

1. An Intra-coded (I) VOP is coded using information only from itself.

2. A Predictive-coded (P) VOP is a VOP which is coded using motion compensated prediction from a pa reference VOP.

3. A Bidirectionally predictive-coded (B) VOP is a VOP which is coded using motion compensated prediction fro a past and/or future reference VOP(s).

4. A sprite (S) VOP is a VOP for a sprite object.

6.1.3.5 I-VOPs and group of VOPs

I-VOPs are intended to assist random access into the sequence. Applications requiring random access, fast-forwarc playback, or fast reverse playback may use I-VOPs relatively frequently.

I-VOPs may also be used at scene cuts or other cases where motion compensation is ineffective.

Group of VOP (GOV) header is an optional header that can be used immediately before a coded I-VOP to indicate to the decoder:

1) the modulo part (i.e. the full second units) of the time base for the next VOP after the GOV header in display order

2) if the first consecutive B-VOPs immediately following the coded I-VOP can be reconstructed properly in the case of a random access.

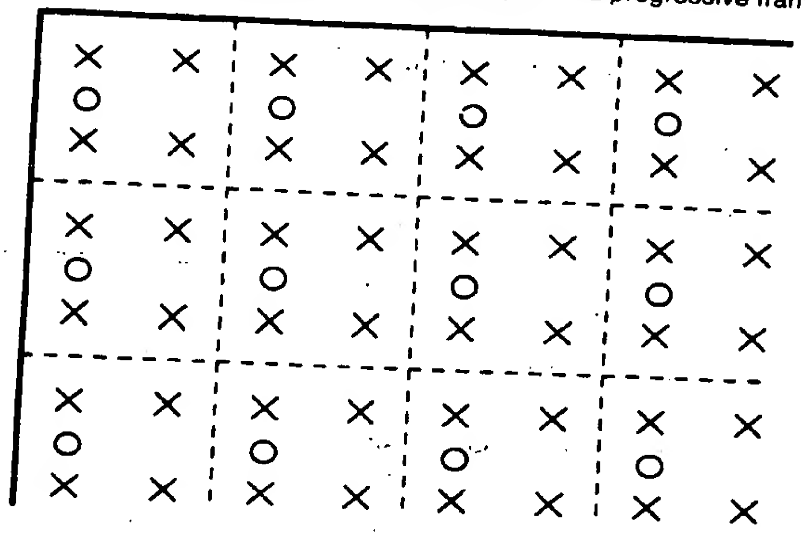
In a non scalable bitstream or the base layer of a scalable bitstream, the first coded VOP following a GOV header shall be a coded I-VOP.

6.1.3.6 Format

In this format the Cb and Cr matrices shall be one half the size of the Y-matrix in both horizontal and vertical dimensions. The Y-matrix shall have an even number of lines and samples.

The luminance and chrominance samples are positioned as shown in Figure 6-1.The two variations in the vertical and temporal positioning of the samples for interlaced VOPs are shown in Figure 6-2 and Figure 6-3.

Figure 6-4 shows the vertical and temporal positioning of the samples in a progressive frame.



× Represent luminance samples

○ Represent chrominance samples

Figure 6-1 – The position of luminance and chrominance samples in 4:2:0 data

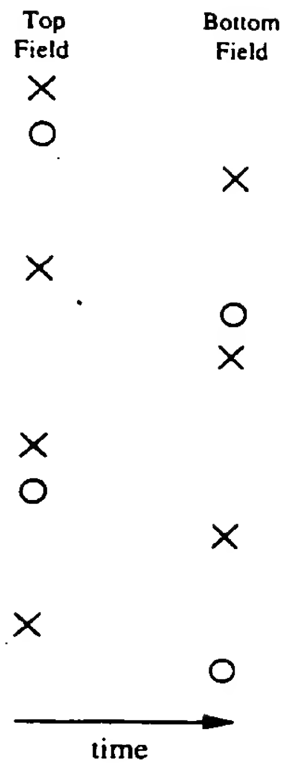


Figure 6-2 -- Vertical and temporal positions of samples in an interlaced frame with top_field_first=1

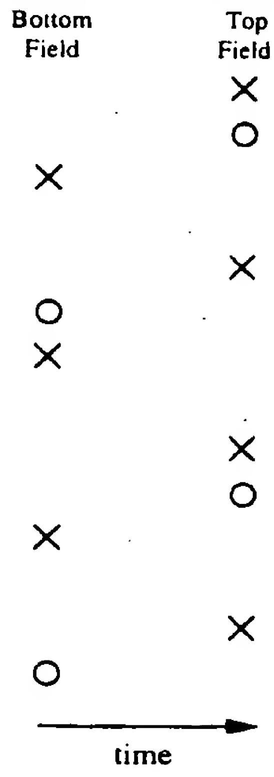


Figure 6-3 -- Vertical and temporal position of samples in an interlaced frame with top_field_first=0

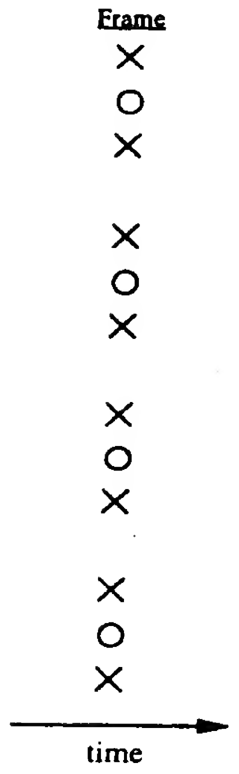


Figure 6-4 -- Vertical and temporal positions of samples in a progressive frame

The binary alpha plane for each VOP is represented by means of a bounding rectangle as described in clause F.2, and it has always the same number of lines and pixels per line as the luminance plane of the VOP bounding rectangle. The positions between the luminance and chrominance pixels of the bounding rectangle are defined in this clause according to the 4:2:0 format. For the progressive case, each 2x2 block of luminance pixels in the bounding rectangle associates to one chrominance pixel. For the interlaced case, each 2x2 block of luminance pixels of the same field in the bounding rectangle associates to one chrominance pixel of that field.

In order to perform the padding process on the two chrominance planes, it is necessary to generate a binary alpha plane which has the same number of lines and pixels per line as the chrominance planes. Therefore, when non-scalable shape coding is used, this binary alpha plane associated with the chrominance planes is created from the binary alpha plane associated with the luminance plane by the subsampling process defined below:

For each 2x2 block of the binary alpha plane associated with the luminance plane of the bounding rectangle (of the same frame for the progressive and of the same field for the interlaced case), the associated pixel value of the binary alpha plane associated with the chrominance planes is set to 255 if any pixel of said 2x2 block of the binary alpha plane associated with the luminance plane equals 255.

6.1.3.7 VOP reordering

When a video object layer contains coded B-VOPs, the number of consecutive coded B-VOPs is variable and unbounded. The first coded VOP shall not be a B-VOP.

A video object layer may contain no coded P-VOPs. A video object layer may also contain no coded I-VOPs in which case some care is required at the start of the video object layer and within the video object layer to effect both random access and error recovery.

The order of the coded VOPs in the bitstream, also called decoding order, is the order in which a decoder reconstructs them. The order of the reconstructed VOPs at the output of the decoding process, also called the display order, is not always the same as the decoding order and this subclause defines the rules of VOP reordering that shall happen within the decoding process.

When the video object layer contains no coded B-VOPs, the decoding order is the same as the display order.

When B-VOPs are present in the video object layer re-ordering is performed according to the following rules:

If the current VOP in decoding order is a B-VOP the output VOP is the VOP reconstructed from that B-VOP.

If the current VOP in decoding order is a I-VOP or P-VOP the output VOP is the VOP reconstructed from the previous I-VOP or P-VOP if one exists. If none exists, at the start of the video object layer, no VOP is output.

The following is an example of VOPs taken from the beginning of a video object layer. In this example there are two coded B-VOPs between successive coded P-VOPs and also two coded B-VOPs between successive coded I- and P-VOPs. VOP '1I' is used to form a prediction for VOP '4P'. VOPs '4P' and '1I' are both used to form predictions for VOPs '2B' and '3B'. Therefore the order of coded VOPs in the coded sequence shall be '1I', '4P', '2B', '3B'. However, the decoder shall display them in the order '1I', '2B', '3B', '4P'.

At the encoder input,

1	2	3	4	5	6	7	8	9	10	11	12	13
I	B	B	P	B	B	P	B	B	I	B	B	P

At the encoder output, in the coded bitstream, and at the decoder input,

1	4	2	3	7	5	6	10	8	9	13	11	12
I	P	B	B	P	B	B	I	B	B	P	B	B

At the decoder output,

1	2	3	4	5	6	7	8	9	10	11	12	13
I	B	B	P	B	B	P	B	B	I	B	B	P

6.1.3.8 Macroblock

A macroblock contains a section of the luminance component and the spatially corresponding chrominance components. The term macroblock can either refer to source and decoded data or to the corresponding coded data elements. A skipped macroblock is one for which no information is transmitted. Presently there is only one chrominance format for a macroblock, namely, 4:2:0 format. The orders of blocks in a macroblock is illustrated below:

A 4:2:0 Macroblock consists of 6 blocks. This structure holds 4 Y, 1 Cb and 1 Cr Blocks and the block order is depicted in Figure 6-5.

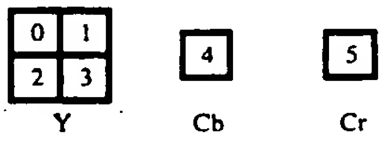


Figure 6-5 – 4:2:0 Macroblock structure

The organisation of VOPs into macroblocks is as follows.

For the case of a progressive VOP, the interlaced flag (in the VOP header) is set to "0" and the organisation of lines of luminance VOP into macroblocks is called frame organization and is illustrated in Figure 6-6. In this case, frame DCT coding is employed.

For the case of interlaced VOP, the interlaced flag is set to "1" and the organisation of lines of luminance VOP into macroblocks can be either frame organization or field organization and thus both frame and field DCT coding may be used in the VOP.

BIFS syntax (see ISO/IEC 14496-1). The FDP node defines the face model to be used at the receiver. Two options are supported:

- calibration information is downloaded so that the proprietary face of the receiver can be configured using facial feature points and optionally a 3D mesh or texture.
- a face model is downloaded with the animation definition of the Facial Animation Parameters. This face model replace the proprietary face model in the receiver.

6.2 Visual bitstream syntax

6.2.1 Start codes

Start codes are specific bit patterns that do not otherwise occur in the video stream.

Each start code consists of a start code prefix followed by a start code value. The start code prefix is a string of twenty three bits with the value zero followed by a single bit with the value one. The start code prefix is thus the bit string '0000 0000 0000 0000 0000 0001'.

The start code value is an eight bit integer which identifies the type of start code. Many types of start code have just one start code value. However video_object_start_code and video_object_layer_start_code are represented by many start code values.

All start codes shall be byte aligned. This shall be achieved by first inserting a bit with the value zero and then, if necessary, inserting bits with the value one before the start code prefix such that the first bit of the start code prefix is the first (most significant) bit of a byte. For stuffing of 1 to 8 bits, the codewords are as follows in Table 6-2.

Table 6-2-- Stuffing codewords

Bits to be stuffed	Stuffing Codeword
1	0
2	01
3	011
4	0111
5	01111
6	011111
7	0111111
8	01111111

Table 6-3 defines the start code values for all start codes used in the visual bitstream.

Table 6-3 — Start code values

name	start code value (hexadecimal)
video_object_start_code	00 through 1F
video_object_layer_start_code	20 through 2F
reserved	30 through AF
visual_object_sequence__start_code	B0

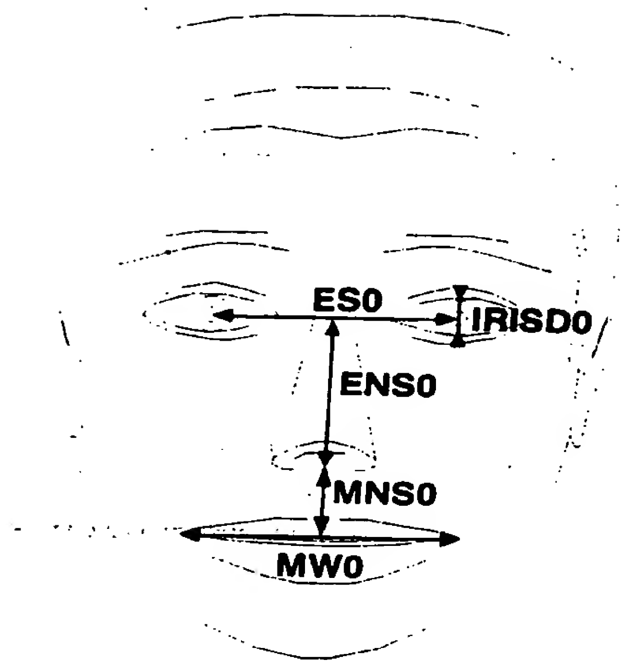


Figure 6-10 – The Facial Animation Parameter Units

6.1.5.4 Description of a neutral face

At the beginning of a sequence, the face is supposed to be in a neutral position. Zero values of the FAPs correspond to a neutral face. All FAPs are expressed as displacements from the positions defined in the neutral face. The neutral face is defined as follows:

- the coordinate system is right-handed; head axes are parallel to the world axes
- gaze is in direction of Z axis
- all face muscles are relaxed
- eyelids are tangent to the iris
- the pupil is one third of IRISD0
- lips are in contact; the line of the lips is horizontal and at the same height of lip corners
- the mouth is closed and the upper teeth touch the lower ones
- the tongue is flat, horizontal with the tip of tongue touching the boundary between upper and lower teeth (feature point 6.1 touching 9.11 in annex C)

6.1.5.5 Facial definition parameter set

The FDPs are used to customize the proprietary face model of the decoder to a particular face or to download a face model along with the information about how to animate it. The definition and description of FDP fields is given in annex C. The FDPs are normally transmitted once per session, followed by a stream of compressed FAPs. However, if the decoder does not receive the FDPs, the use of FAPUs ensures that it can still interpret the FAP stream. This insures minimal operation in broadcast or teleconferencing applications. The FDP set is specified in

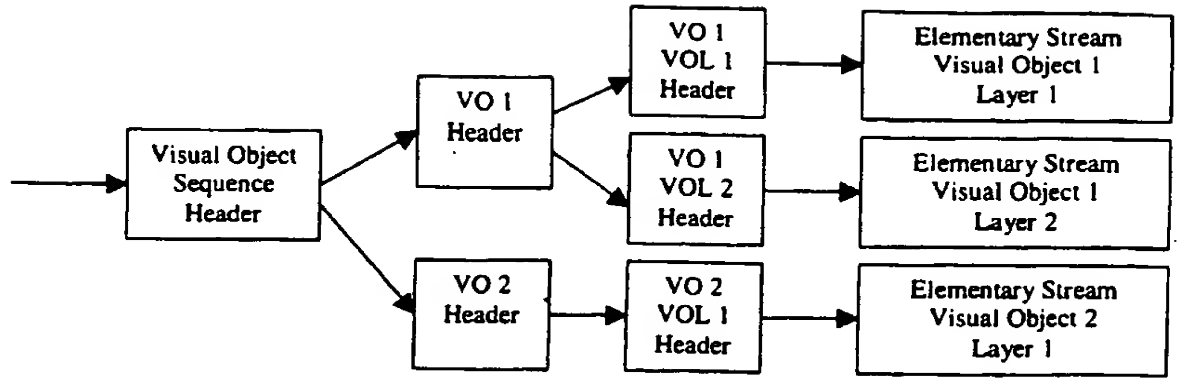

Visual Object Sequence Header
VO 1 Header
VO 2 Header
VO 1 VOL 1 Header
VO 1 VOL 2 Header
VO 2 VOL 1 Header
Elementary Stream Visual Object 1 Layer 1
Elementary Stream Visual Object 1 Layer 2
Elementary Stream Visual Object 2 Layer 1

Figure 6-11 – Example Visual Information – Logical Structure

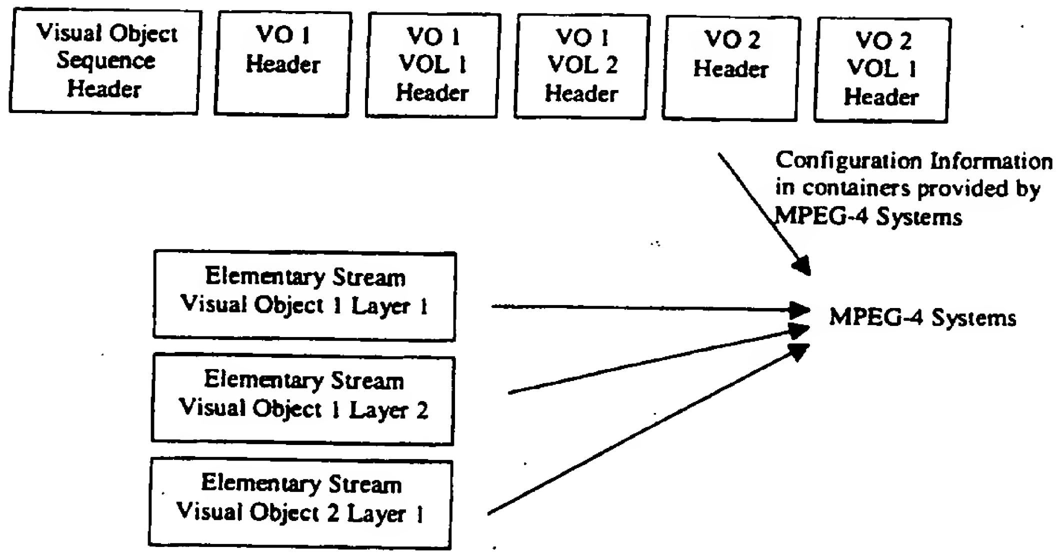

Visual Object Sequence Header
VO 1 Header
VO 1 VOL 1 Header
VO 1 VOL 2 Header
VO 2 Header
VO 2 VOL 1 Header
Configuration Information in containers provided by MPEG-4 Systems
Elementary Stream Visual Object 1 Layer 1
Elementary Stream Visual Object 1 Layer 2
Elementary Stream Visual Object 2 Layer 1
MPEG-4 Systems

Figure 6-12 – Example Visual Bitstream – Separate Configuration Information / Elementary Stream.

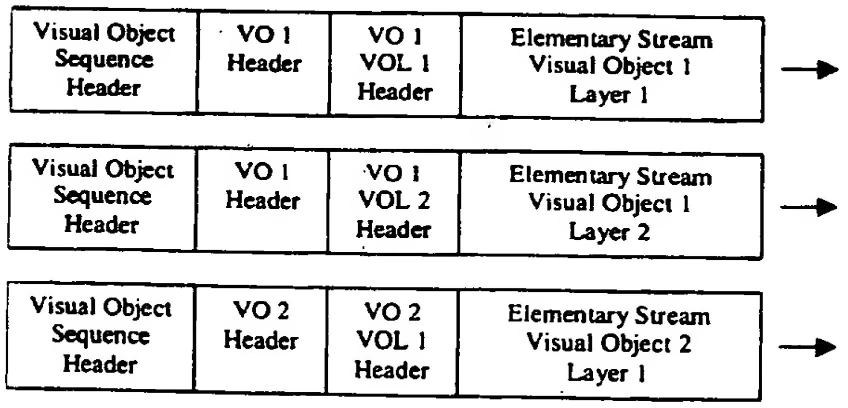

Visual Object Sequence Header
VO 1 Header
VO 1 VOL 1 Header
Elementary Stream Visual Object 1 Layer 1
Visual Object Sequence Header
VO 1 Header
VO 1 VOL 2 Header
Elementary Stream Visual Object 1 Layer 2
Visual Object Sequence Header
VO 2 Header
VO 2 VOL 1 Header
Elementary Stream Visual Object 2 Layer 1

Figure 6-13 – Example Visual Bitstream – Combined Configuration Information / Elementary Stream

The following functions are entry points for elementary streams, and entry into these functions defines the breakpoint between configuration information and elementary streams:

1. Group_of_VideoObjectPlane(),

2. VideoObjectPlane().
3. video_plane_with_short_header().
4. MeshObject().
5. FaceObject().

For still texture objects, configuration information ends and elementary stream data begins in StilTextureObject() immediately before the first call to wavelet_dc_decode(), as indicated by the comment in subclause 6.2.8.

There is no overlap of syntax between configuration information and elementary streams.

The configuration information contains all data that is not part of an elementary stream, including that defined by VisualObjectSequence(), VisualObject() and VideoObjectLayer().

ISO/IEC 14496-2 does not provide for the multiplexing of multiple elementary streams into a single bitstream. One visual bitstream contains exactly one elementary stream, which describes one layer of one visual object. A visual decoder must conceptually have a separate entry port for each layer of each object to be decoded.

Visual objects coded in accordance with this Part may be carried within a Systems bitstream as defined by ISO/IEC 14496-1. The coded visual objects may also be free standing or carried within other types of systems. Configuration information may be carried separately from or combined with elementary stream data:

1. *Separate Configuration / Elementary Streams (e.g. Inside ISO/IEC 14496-1 Bitstreams)*

When coded visual objects are carried within a Systems bitstream defined by ISO/IEC 14496-1, configuration information and elementary stream data are always carried separately. Configuration information and elementary streams follow the syntax below, subject to the break points between them defined above. The Systems specification ISO/IEC 14496-1 defines containers that are used to carry Visual Object and Visual Object Layer configuration information. A separate container is used for each object. For video objects, a separate container is also used for each layer. VisualObjectSequence headers are not carried explicitly, but the information is contained in other parts of the Systems bitstream.

2. *Combined Configuration / Elementary Streams*

The elementary stream data associated with a single layer may be wrapped in configuration information defined in accordance with the syntax below. A visual bitstream may contain at most one instance of each of VisualObjectSequence(), VisualObject() and VideoObjectLayer(). The Visual Object Sequence Header must be identical for all streams input simultaneously to a decoder. The Visual Object Headers for each layer of a multilayer object must be identical.

6.2.2 Visual Object Sequence and Visual Object

	No. of bits	Mnemonic
VisualObjectSequence() {		
visual_object_sequence_start_code	32	bslbf
profile_and_level_indication	8	uimsbf
while (next_bits()== user_data_start_code){		
user_data()		
}		
VisualObject()		
visual_object_sequence_end_code	32	bslbf
}		

	No. of bits	Mnemonic
VisualObject() {		
visual_object_start_code	32	bslbf

is_visual_object_identifier	1	uimsbf
if (is_visual_object_identifier) {		
visual_object_verid	4	uimsbf
visual_object_priority	3	uimsbf
}		
visual_object_type	4	uimsbf
if (visual_object_type == "video ID" II visual_object_type == "still texture ID") {		
video_signal_type()		
}		
next_start_code()		
while (next_bits()== user_data_start_code){		
user_data()		
}		
if (visual_object_type == "video ID") {		
video_object_start_code	32	bslbf
VideoObjectLayer()		
}		
else if (visual_object_type == "still texture ID") {		
StillTextureObject()		
}		
else if (visual_object_type == "mesh ID") {		
MeshObject()		
}		
else if (visual_object_type == "face ID") {		
FaceObject()		
}		
if (next_bits() != "0000 0000 0000 0000 0000 0001")		
next_start_code()		
}		

video_signal_type() {	**No. of bits**	**Mnemonic**
video_signal_type	1	bslbf
if (video_signal_type) {		
video_format	3	uimsbf
video_range	1	bslbf

colour_description	1	bslbf
if (colour_description) {		
colour_primaries	8	uimsbf
transfer_characteristics	8	uimsbf
matrix_coefficients	8	uimsbf
}		
}		
}		

6.2.2.1 User data

	No. of bits	Mnemonic
user_data() {		
user_data_start_code	32	bslbf
while(next_bits() != '0000 0000 0000 0000 0000 0001') {		
user_data	8	uimsbf
}		
next_start_code()		
}		

6.2.3 Video Object Layer

	No. of bits	Mnemonic
VideoObjectLayer() {		
if(next_bits() == video_object_layer_start_code) {		
short_video_header = 0		
video_object_layer_start_code	32	bslbf
random_accessible_vol	1	bslbf
video_object_type_indication	8	uimsbf
is_object_layer_identifier	1	uimsbf
if (is_object_layer_identifier) {		
video_object_layer_verid	4	uimsbf
video_object_layer_priority	3	uimsbf
}		
aspect_ratio_info	4	uimsbf
if (aspect_ratio_info == "extended_PAR") {		
par_width	8	uimsbf
par_height	8	uimsbf
}		

vol_control_parameters	1	bslbf
if (vol_control_parameters) {		
chroma_format	2	uimsbf
low_delay	1	uimsbf
vbv_parameters	1	blsbf
if (vbv_parameters) {		
first_half_bit_rate	15	uimsbf
marker_bit	1	bslbf
latter_half_bit_rate	15	uimsbf
marker_bit	1	bslbf
first_half_vbv_buffer_size	15	uimsbf
marker_bit	1	bslbf
latter_half_vbv_buffer_size	3	uimsbf
first_half_vbv_occupancy	11	uimsbf
marker_bit	1	blsbf
latter_half_vbv_occupancy	15	uimsbf
marker_bit	1	blsbf
}		
}		
video_object_layer_shape	2	uimsbf
marker_bit	1	bslbf
vop_time_increment_resolution	16	uimsbf
marker_bit	1	bslbf
fixed_vop_rate	1	bslbf
if (fixed_vop_rate)		
fixed_vop_time_increment	1-16	uimsbf
if (video_object_layer_shape != "binary only") {		
if (video_object_layer_shape == "rectangular") {		
marker_bit	1	bslbf
video_object_layer_width	13	uimsbf
marker_bit	1	bslbf
video_object_layer_height	13	uimsbf
marker_bit	1	bslbf
}		
interlaced	1	bslbf

obmc_disable	1	bslbf
sprite_enable	1	bslbf
if (sprite_enable) {		
sprite_width	13	uimsbf
marker_bit	1	bslbf
sprite_height	13	uimsbf
marker_bit	1	bslbf
sprite_left_coordinate	13	simsbf
marker_bit	1	bslbf
sprite_top_coordinate	13	simsbf
marker_bit	1	bslbf
no_of_sprite_warping_points	6	uimsbf
sprite_warping_accuracy	2	uimsbf
sprite_brightness_change	1	bslbf
low_latency_sprite_enable	1	bslbf
}		
not_8_bit	1	bslbf
if (not_8_ bit) {		
quant_precision	4	uimsbf
bits_per_pixel	4	uimsbf
}		
if (video_object_layer_shape=="grayscale") {		
no_gray_quant_update	1	bslbf
composition_method	1	bslbf
linear_composition	1	bslbf
}		
quant_type	1	bslbf
if (quant_type) {		
load_intra_quant_mat	1	bslbf
if (load_intra_quant_mat)		
intra_quant_mat	8*[2-64]	uimsbf
load_nonintra_quant_mat	1	bslbf
if (load_nonintra_quant_mat)		
nonintra_quant_mat	8*[2-64]	uimsbf
if(video_object_layer_shape=="grayscale") {		
load_intra_quant_mat_grayscale	1	bslbf

if(load_intra_quant_mat_grayscale)		
intra_quant_mat_grayscale	8*[2-64]	uimsbf
load_nonintra_quant_mat_grayscale	1	bslbf
if(load_nonintra_quant_mat_grayscale)		
nonintra_quant_mat_grayscale	8*[2-64]	uimsbf
}		
}		
complexity_estimation_disable	1	bslbf
if (!complexity_estimation_disable)		
define_vop_complexity_estimation_header()		
resync_marker_disable	1	bslbf
data_partitioned	1	bslbf
if(data_partitioned)		
reversible_vlc	1	bslbf
scalability	1	bslbf
if (scalability) {		
hierarchy_type	1	bslbf
ref_layer_id	4	uimsbf
ref_layer_sampling_direc	1	bslbf
hor_sampling_factor_n	5	uimsbf
hor_sampling_factor_m	5	uimsbf
vert_sampling_factor_n	5	uimsbf
vert_sampling_factor_m	5	uimsbf
enhancement_type	1	bslbf
}		
}		
else		
resync_marker_disable	1	bslbf
next_start_code()		
while (next_bits()== user_data_start_code){		
user_data()		
}		
if (sprite_enable && !low_latency_sprite_enable)		
VideoObjectPlane()		
do {		

if (next_bits() == group_of_vop_start_code)		
Group_of_VideoObjectPlane()		
VideoObjectPlane()		
} while ((next_bits() == group_of_vop_start_code) || (next_bits() == vop_start_code))		
} else {		
short_video_header = 1		
do {		
video_plane_with_short_header()		
} while(next_bits() == short_video_start_marker)		
}		
}		

	No. of bits	Mnemonic
define_vop_complexity_estimation_header() {		
estimation_method	2	uimsbf
if (estimation_method =='00'){		
shape_complexity_estimation_disable	1	bslbf
if (!shape_complexity_estimation_disable) {		
opaque	1	bslbf
transparent	1	bslbf
intra_cae	1	bslbf
inter_cae	1	bslbf
no_update	1	bslbf
upsampling	1	bslbf
}		
texture_complexity_estimation_set_1_disable	1	bslbf
if (!texture_complexity_estimation_set_1_disable) {		
intra_blocks	1	bslbf
inter_blocks	1	bslbf
inter4v_blocks	1	bslbf
not_coded_blocks	1	bslbf
}		
marker_bit	1	bslbf
texture_complexity_estimation_set_2_disable	1	bslbf
if (!texture_complexity_ estimation_set_2_disable) {		